Scalable Betweenness Centrality Maximization via Sampling

Ahmad Mahmoody
ahmad@cs.brown.edu

Charalampos E. Tsourakakis
babis@seas.harvard.edu

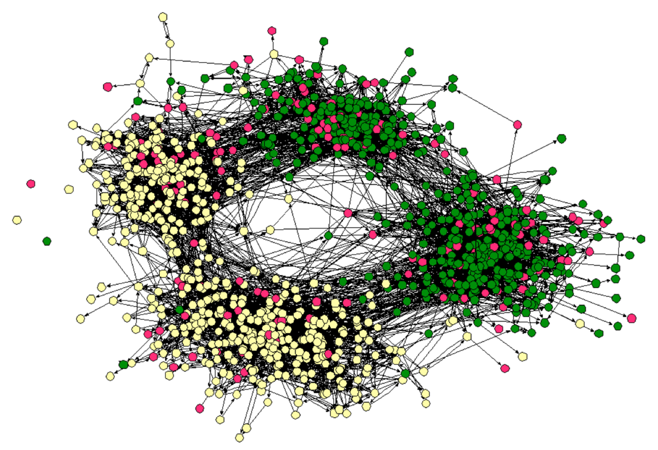
Eli Upfal
eli@cs.brown.edu

Motivation

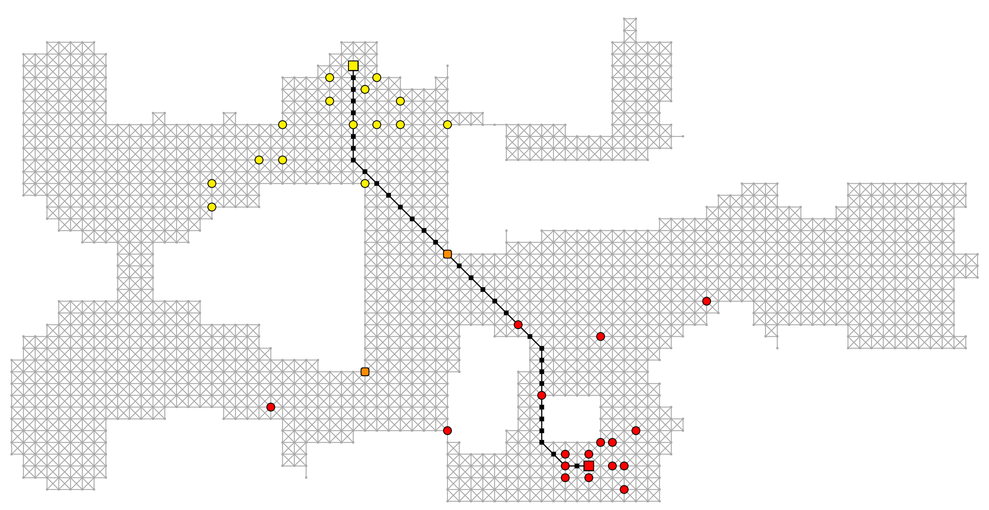
The betweenness centrality of a node u is defined as

$$B(u) = \sum_{s,t} \frac{\sigma_{s,t}(u)}{\sigma_{s,t}},$$

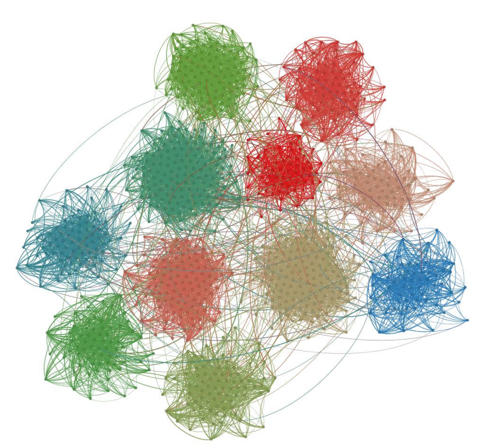
where $\sigma_{s,t}$ is the number of s-t shortest paths, and $\sigma_{s,t}(u)$ is the number of s-t shortest paths that have u as their internal node.



- Community detection: Betweenness centrality is frequently used to detect communities in large scale networks [3].



- Navigation applications: It is also used as a successful heuristic for selecting landmarks in state-of-the-art shortest path applications [1]



- Attacking graph connectivity: Real-world networks are robust to random failures but fragile with respect to targeted attacks. Betweenness centrality is used as a good heuristic to destroy connectivity

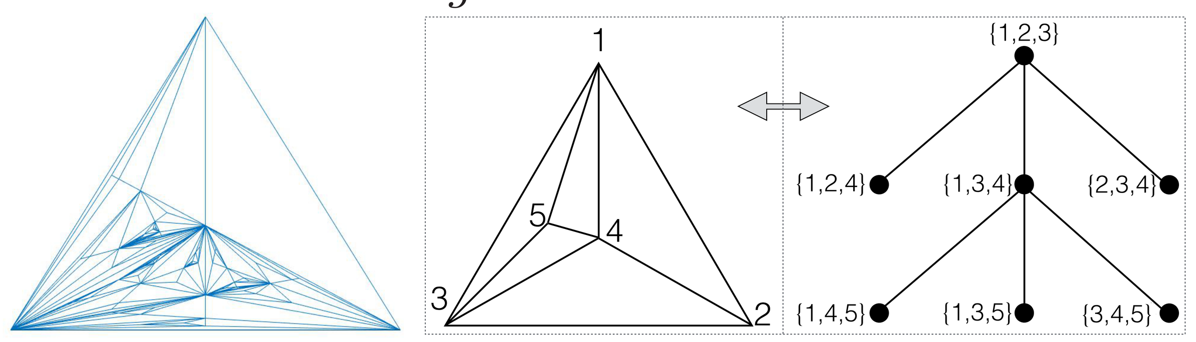
Main contributions

For $S \subseteq V$, we define the betweenness centrality of S as

$$B(S) = \sum_{s,t \in V} \frac{\sigma_{s,t}(S)}{\sigma_{s,t}},$$

where $\sigma_{s,t}(S)$ is the number of s-t shortest paths that have an internal node in S.

Contribution 1 *Prior work on BWC estimation strongly relies on the assumption that* $OPT_k = \Theta(n^2)$ *for a constant integer* k *[4]. We show this assumption is not true in general.*



(a) (b)

We explain empirical evidence which supports this strong assumption using Random Apollonian Networks that provably generate scale-free, small-world graphs with high probability [2]. Also, bounded-tree width networks including Barabasi-Albert random graphs satisfy this assumption.

Contribution 2 *We design* **HEDGE**– *a* $(1-1/e-\epsilon)$*-approximation algorithm – that uses smaller sized samples compared to state-of-the-art [4].*

Algorithm 1: HEDGE

Input: A hyper-edge sampler $\mathcal{A}$ for BWC, number of hyper-edges q, and the size of the output set k.
Output: A subset of nodes, S of size k.

```
begin
    H ← ∅;
    for i ∈ [q] do
        h ~ A (sample a random hyper-edge);
        H ← H ∪ {h};
    S ← ∅ ;
    while |S| < k do
        u ← arg max_{v∈V} deg_H({v});
        S ← S ∪ {u};
        for h ∈ H such that u ∈ h do
            H ← H \ {h};
    return S;
```

Contribution 3 *We provide a general analytical framework based on Chernoff bound and submodular optimization, and show that it can be applied to any other centrality measure if it (i) is monotone-submodular, and (ii) admits a hyper-edge sampler*

Experimental Results

GRAPHS	#nodes	#edges	k	Algorithms: EXHAUST	Algorithms: HEDGE	speedup
ca-GrQd	5242	14496	10	0.242	0.241	2.616
			50	0.713	0.699	2.516
			100	0.974	0.951	2.217
p2p-Gnutella08	6301	20777	10	0.013	0.011	6.773
			50	0.036	0.035	6.478
			100	0.053	0.051	6.117
ca-HepTh	9877	25998	10	0.165	0.164	4.96
			50	0.498	0.497	4.729
			100	0.747	0.745	4.473

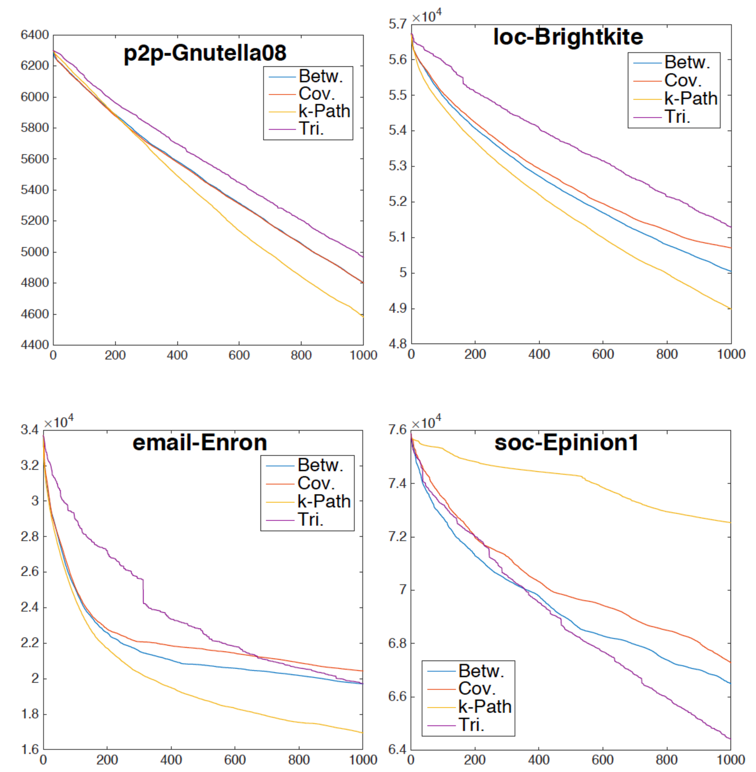
HEDGE vs. EXHAUST (baseline method): centralities and speedups.

		Betw. Centrality			# of Samples		
GRAPHS	k	Y-ALG	HEDGE$_=$	HEDGE	Y-ALG	HEDGE$_=$	HEDGE
CA-GrQc	10	0.208	0.214	0.215	5278		8565
	50	0.484	0.483	0.49			42822
	100	0.569	0.568	0.577			85643
CA-HepTh	10	0.151	0.151	0.154	5658		9198
	50	0.403	0.4	0.409			45989
	100	0.534	0.533	0.547			91978
ego-Facebook	10	0.924	0.932	0.933	5121		8304
	50	0.959	0.957	0.959			41519
	100	0.962	0.96	0.964			83038
email-Enron	10	0.329	0.335	0.335	6445		10511
	50	0.644	0.646	0.65			52552
	100	0.754	0.756	0.762			105104

Our proposed method outperforms the state-of-the-art method due to Yoshida [4]

		METHODS				
GRAPHS	k	IM	betw.	cov.	κ-path	tri.
CA-GrQc	10	19.12	13.67	14.93	14.10	18.48
	50	76.65	67.28	67.44	65.06	69.30
	100	141.33	126.76	126.66	124.51	124.06
CA-HepTh	10	17.33	15.61	15.58	14.63	12.98
	50	77.88	70.53	69.95	67.80	63.95
	100	147.75	133.45	133.24	130.41	127.52
p2p-Gnutella08	10	19.61	13.05	13.71	10.39	18.06
	50	83.64	60.58	61.73	51.57	74.19
	100	148.86	118.27	118.76	103.58	132.04
email-Enron	10	461.84	458.70	450.34	455.25	451.53
	50	719.86	703.08	695.81	699.74	681.05
	100	887.63	863.66	858.39	865.76	830.15
loc-Brightkite	10	184.40	162.64	160.35	163.16	145.19
	50	402.85	372.64	360.64	366.28	330.45
	100	563.13	521.18	508.59	512.77	445.11
soc-Epinion1	10	343.89	81.57	111.47	14.43	311.74
	50	846.18	300.88	282.88	72.90	778.56
	100	1161.45	463.04	457.29	133.20	1062.99

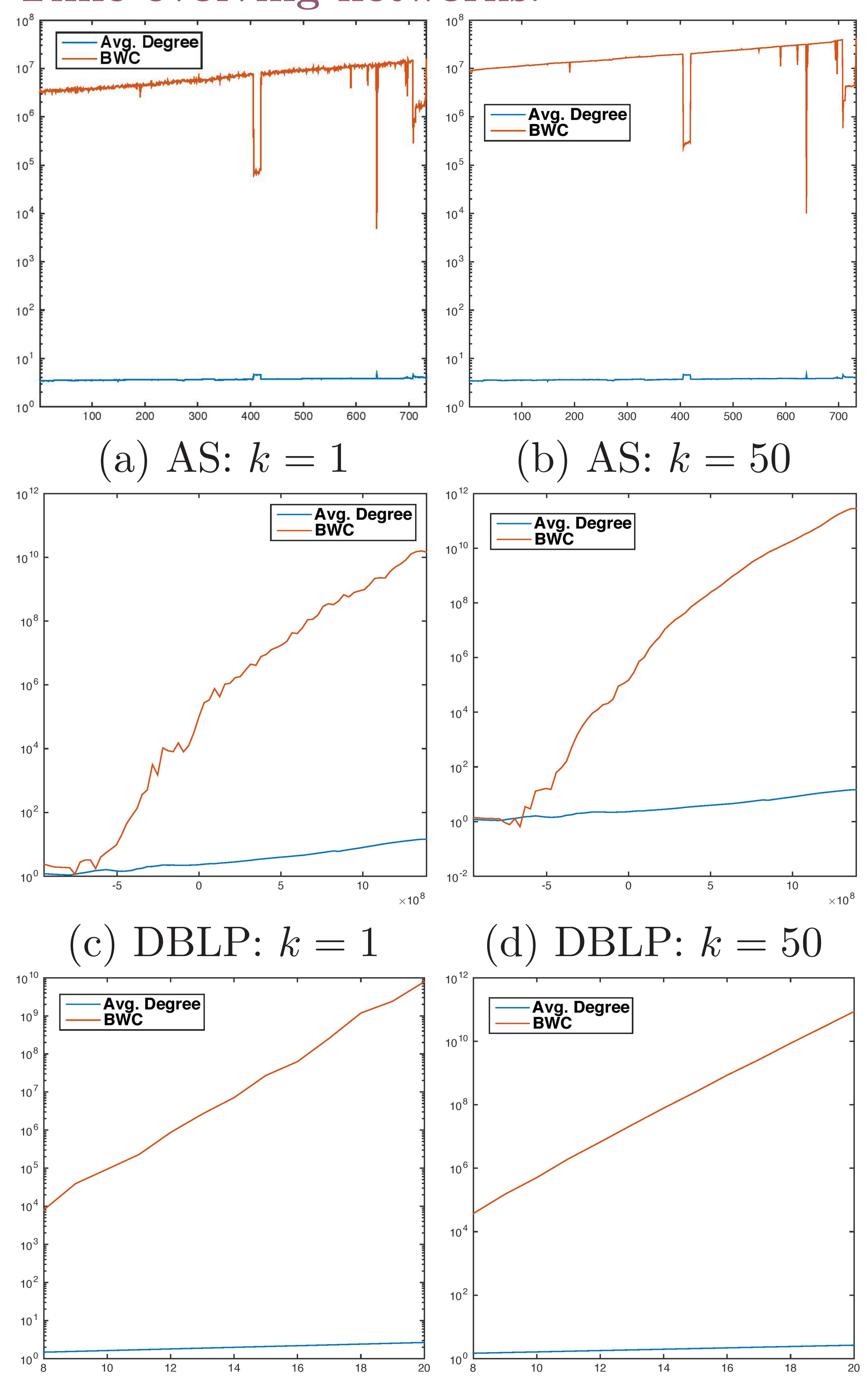
Our proposed algorithm can be used to scale heuristic uses of BWC for influence maximization.



The size of the largest connected component, as we remove the first 1000 nodes in the order induced by centralities.

Experimental results

Time evolving networks:



(a) AS: $k=1$ (b) AS: $k=50$

(c) DBLP: $k=1$ (d) DBLP: $k=50$

(e) KG: $k=1$ (f) KG: $k=50$

Largest betweenness centrality score and number of nodes, edges and average degree versus time on the (i) Autonomous systems (a),(b) (ii) DBLP dataset (c),(d) and (iii) stochastic Kronecker graphs (e),(f).

References

[1] I. Abraham, D. Delling, A. Goldberg, R. Werneck. Hierarchical hub labelings for shortest paths. *ESA 2012*

[2] A. Frieze and C. E. Tsourakakis. Some properties of random apollonian networks. *Internet Mathematics*, 10(1-2):162–187, 2014.

[3] M. Girvan and M. E. Newman. Community structure in social and biological networks. *PNAS*, 99(12):7821–7826, 2002.

[4] Y. Yoshida. Almost linear-time algorithms for adaptive betweenness centrality using hypergraph sketches. KDD, 2014.